technische universität dortmund

Prof. Dr. Jan Jürjens, Christian Wessel
Lehrstuhl 14 – Software Engineering
Fakultät für Informatik
Modellbasierte Softwaretechniken
für sichere Systeme – 15.05.2012

Modellbasierte Softwaretechniken für sichere Systeme - Übung 5

1 Secure Dependencies

1.1 Erklären Sie die Unterschiede zwischen Secure Links und Secure Dependencies (1 Punkt)

Geben Sie jeweils ein Beispiel dazu an.

1.2 Modellierung von Secure Dependencies (5 Punkte)

Folgendes Szenario sei gegeben:

Ein Bankkunde fordert ein neue TANListe an. Das PrintSystem der Bank braucht daher eine neue TAN-Liste, die es ausdrucken kann. Dazu wird der Server vom PrintSystem mit der Erstellung der TAN-Liste beauftragt. Der Vorgang muss immer funktionieren und die Daten dürfen dabei auf keinen Fall verändert werden, da ansonsten rechtliche Probleme auftreten.
Der Server benötigt zur Listenerstellung einen Zufallszahlengenerator. Dabei ist zu beachten, dass die generierte Zufallszahl in jedem Fall geheim bleibt.
Der Server erstellt ein Log dieser Transaktion. Diese Daten sind zwar nicht sicherheitskritisch, jedoch soll verhindert werden, dass sie nachträglich verändert werden dürfen.
Außerdem können Kunden Nachrichten an ihre Kundenberater schicken, welche verschlüsselt im System gespeichert werden. Dieses können sie entweder über eine Handyapplikation oder über eine normale Applikation vom PC durchführen Da die das Signieren der Nachricht zu viel Rechenkraft erfordern würde und der Akku des Handys geschont werden soll, werden die Nachrichten unsigniert versendet. Es sollen jedoch sichergestellt sein, dass die Nachrichten nicht verfälscht werden können Die normale Desktop-Applikation signiert die Nachrichten vor dem Versenden.

Modellieren Sie das Klassendiagramm des Banksystems und ordnen Sie den Schutzbedarf der Methoden entsprechend ein.

Benutzen Sie dafür folgende Klassen- und Methodennamen:

- ServerCore
  - writeLog(String msg):void
  - generateTANList():TanList

  - ~~getTANListForUser(String userId):void~~
  - encryptMessage(String msg):String

- PrintSystem
  - printTANList(String userId):void
- RandomNumberGenerator
  - generate():int
- LoggingSystem
  - log(String msg):void
- MessagingSystem
  - sendMessage(String msg):void
- SignedMessaging
  - sendSignedMessage():void

2 Guarded Objects (2 Punkte)

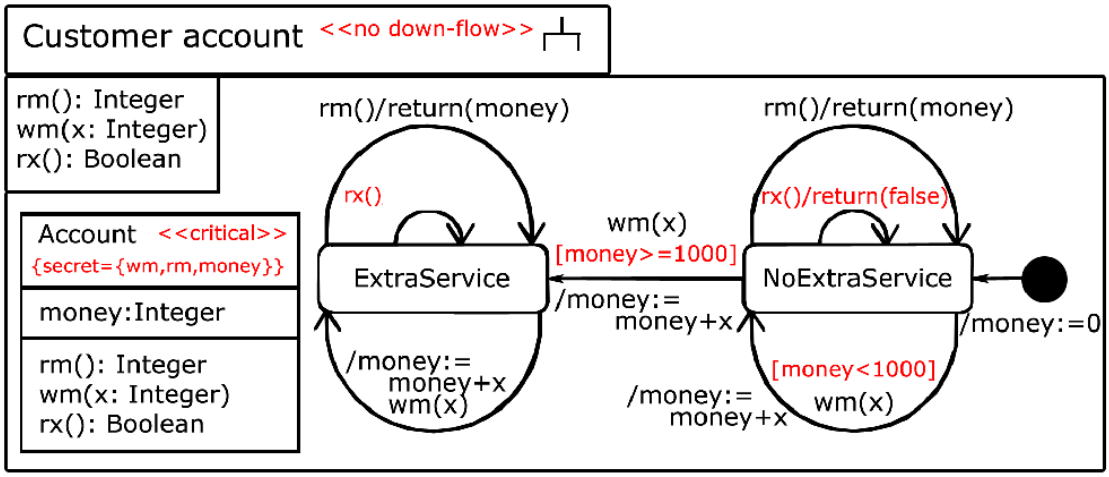
Die Klasse `java.security.GuardedObject` schützt den Zugang zu anderen Objekten. Auf das geschützte Objekt kann über die `getObject`-Methode zugegriffen werden. Die Methode `checkGuard`-Methode des Interface `Guard` kontrolliert dabei den Zugang zum geschützten Objekt und gibt eine Referenz oder eine SecurityException zurück.

2.1 Warum ist dieser Mechanismus nicht völlig sicher?

2.2 Wie könnte man ihn erweitern, um dieses Problem zu begrenzen?

3 <<no down-flow>> (2 Punkte)

Im Zustand ExtraService wird auf rx() kein Rückgabewert zurückgegeben. Ist <<no down-flow>> nun erfüllt (mit Begründung)?


Customer account
<<no down-flow>>
rm(): Integer
wm(x: Integer)
rx(): Boolean
Account <<critical>>
{secret={wm,rm,money}}
money:Integer
rm(): Integer
wm(x: Integer)
rx(): Boolean
rm()/return(money)
rx()
ExtraService
/money:=
money+x
wm(x)
wm(x)
[money>=1000]
/money:=
money+x
rm()/return(money)
rx()/return(false)
NoExtraService
[money<1000]
/money:=
money+x
wm(x)
/money:=0